ASCII Character Codes

DEC	CHAR	DEC	CHAR	DEC	CHAR
000	CTRL-@	043	+	086	V
001	CTRL-A	044	,	087	W
002	CTRL-B	045	–	088	X
003	CTRL-C	046	.	089	Y
004	CTRL-D	047	/	090	Z
005	CTRL-E	048	0	091	[
006	CTRL-F	049	1	092	\
007	CTRL-G	050	2	093	]
008	BS	051	3	094	↑
009	HT	052	4	095	—
010	LF	053	5	096	`
011	VT	054	6	097	a
012	FF	055	7	098	b
013	CR	056	8	099	c
014	CTRL-N	057	9	100	d
015	CTRL-O	058	:	101	e
016	CTRL-P	059	;	102	f
017	CTRL-Q	060	<	103	g
018	CTRL-R	061	=	104	h
019	CTRL-S	062	>	105	i
020	CTRL-T	063	?	106	j
021	CTRL-U	064	@	107	k
022	CTRL-V	065	A	108	l
023	CTRL-W	066	B	109	m
024	CTRL-X	067	C	110	n
025	CTRL-Y	068	D	111	o
026	CTRL-Z	069	E	112	p
027	CTRL-[	070	F	113	q
028	CTRL-\	071	G	114	r
029	CTRL-]	072	H	114	s
030	CTRL-↑	073	I	116	t
031	CTRL-_	074	J	117	u
032	SPACE	075	K	118	v
033	!	076	L	119	w
034	"	077	M	120	x
035	#	078	N	121	y
036	$	079	O	122	z
037	%	080	P	123	{
038	&	081	Q	124	l
039	'	082	R	125	}
040	(	083	S	126	~
041	)	084	T	127	DEL
042	★	085	U		

BS = Backspace
CR = Carriage Return
CTRL = Control Character
DEL = Rubout
FF = Form Feed
HT = Horizontal Tab
LF = Line Feed
VT = Vertical Tab

Cromemco RATFOR Features

- **RAT**ional **FOR**tran is a Fortran IV preprocessor that gives the Fortran IV user the features and control structures needed to write structured programs that are readily debugged and modified:

break	**include**
define	**next**
do	null statement (;)
for	**repeat . . . until**
if . . . else	**while**

- Braces, { }, enclosing a group of statements cause those statements to be treated as a single block.
- Control structures, conditionals, and statements can be nested within each other to a maximum level of 100.
- RATFOR recognizes the following logical symbols:

= =	(equal to)
^ =	(not equal to)
<	(less than)
< =	(less than or equal to)
>	(greater than)
> =	(greater than or equal to)
&	(and)
|	(or)
^	(not)

- RATFOR extends Fortran IV without changing any of its features — the user can continue to use all of Fortran's power while writing more easily understood programs.
- Programs are written in RATFOR, translated by RATFOR into Fortran IV at up to 200 lines per minute, then compiled using Cromemco's Fortran IV compiler.
- The RATFOR package includes:
  the RATFOR preprocessor and support programs supplied on 8" or 5" disk;
  the RATFOR reference manual with examples and user instructions;
  a copy of the RATFOR tutorial, **Software Tools**, by Kernighan and Plauger;
  Cromemco's fast Fortran IV compiler with Link, the linking loader, supplied on a second 8" or two additional 5" disks;
  Cromemco's clearly written Fortran IV manual.

Copyright©1979 Cromemco Inc. All rights reserved.

Cromemco incorporated
Tomorrow's Computers Today
280 BERNARDO AVE. MOUNTAIN VIEW, CA 94043

Part No. 023-0082
September 1979

Cromemco RATFOR

RATional FORtran-a preprocessor that brings structured program design to the Fortran IV user

Program control structures

for, **repeat . . . until**, and **while** provide loop control with before or after testing to simplify program logic

if . . . else allows straightforward programming of conditional tests

Compound statements and control structure nesting clarify the meanings of routines

Readability and source file inclusion

define allows the programmer to use meaningful symbols in place of constants

include allows existing RATFOR source modules to be read from disk and inserted into the RATFOR program at preprocessing time

Full Fortran IV capability

All features of Cromemco's Fortran IV are available for use in RATFOR programs

YOUR LOCAL DEALER IS

Cromemco RATFOR Elements

Notation

[ ] Items enclosed within brackets are optional.

Logical symbols

These RATFOR symbols are translated into Fortran logical and relational operators.

RATFOR symbol	Fortran operator
= =	.EQ.
^ =	.NE.
<	.LT.
< =	.LE.
>	.GT.
> =	.GE.
&	.AND.
|	.OR.
^	.NOT.

RATFOR statements

break

causes control to exit a **do**, **for**, **repeat**, or **while** loop and continue at the first statement following the loop;

define (symbol, replacement string)

each occurrence of the symbol in the program or **include**d file is replaced by the replacement string, which is then processed;
symbol and *string* may each contain 1-200 characters;

do index = start, limit [, increment] statement

the RATFOR **do** is identical in function (although not in form) to the Fortran IV DO loop;

for ([initialize] ; [condition] ; [reinitialize]) statement

initialize is executed, then *statement* and *reinitialize* are executed as long as *condition* is true;
condition is tested **before** each iteration;
initialize and *reinitialize* are single Fortran IV statements;
omitting *condition* yields an infinite loop;

if (condition)
statement-1
[**else**
statement-2]

when *condition* is true, *statement-1* is executed;
when *condition* is false, *statement-2* is executed;

include filename

the contents of the file with *filename* are read in as source and processed; the file must be on the same disk as the primary input file;

next

the rest of the containing loop is skipped and execution continues with the next iteration of the loop;

do, **repeat . . . until**, and **while**:
control proceeds to the condition test;

for:
control proceeds to the reinitialize statement;

infinite **repeat**:
control proceeds to the top of the loop;

null statement —
; (used by itself)

; may be used anywhere that another RATFOR statement may be used;

repeat
statement
[**until** (condition)]

statement is executed as long as *condition* is true;
condition is tested **after** each iteration;
omitting the **until** part yields an infinite loop;

while (condition)
statement

statement is executed as long as *condition* is true;
condition is tested **before** each iteration;

RATFOR features

comments

A sharp sign (#) used anywhere on a line causes the rest of the line to be treated as a comment, unless the # is part of a quoted literal.

compound statements

Braces, { }, can be used to enclose single or multiple RATFOR or Fortran statements so that the enclosed block of statements may be used anywhere that a single RATFOR statement may be used.

conditions

Conditions are formed by separating two data items or defined symbols with one of the RATFOR symbols = =, ^ =, <, < =, >, or > =. Multiple conditions are formed by connecting two RATFOR conditions with one of the logical operators | or &.

indented output

The Fortran statements in the generated output are successively indented for each level of nesting in the RATFOR input.

continuation lines

RATFOR generates Fortran continuation lines when multiple conditions extend to the next line, and when a line ends with a comma, as in a long FORMAT statement.

literals

RATFOR recognizes a string of characters enclosed within matching single or double quotes as a quoted literal. RATFOR does not recognize Hollerith literals.

names

Names used in a RATFOR program may consist of 1-6 alphanumeric characters, the first of which must be a letter.

nested statements

RATFOR statements may be nested within other RATFOR statements to a maximum level of 100.

RATFOR execution

Instructions are given to RATFOR using a command line:

[output name] [,listing option] = input name

The file names have the form:

drive:name.extension

Drive defaults to the current drive.
The input name is required.
The input extension defaults to "RFR".
The output name defaults to the input name.
The output extension defaults to "FOR".

The listing options are:

TTY: to direct the listing to the terminal,
PRT: to direct the listing to the printer, and
<nothing> to suppress the listing.